

01/20/99
JCS20 U.S. PRO
ARNOLD
WHITE &
DURKEE
A PROFESSIONAL CORPORATION
ATTORNEYS AT LAW

Austin
Chicago
Houston
Menlo Park
Minneapolis
Washington


750 Bering Drive
Houston, Texas 77057-2198

Telephone 713.787.1400
Facsimile 713.787.1440

D.C. Toedt III
Direct Line 713.787.1408

January 20, 1999

FILE: BVEW:154

CERTIFICATE OF EXPRESS MAILING
NUMBER EI 371 159 796 US
DATE OF DEPOSIT January 20, 1999
I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington, DC 20231.

Signature

BOX PATENT APPLICATION
Assistant Commissioner for Patents
Washington, DC 20231

RE: *U.S. Patent Application Entitled: "Software-Implemented Method for Identifying Nodes on a Network," by Scott H. Hutchinson and Gregory M. Hanka*

Sir:

Transmitted herewith for filing are:

- (1) 35-page patent specification with 20 claims and an abstract (also Figures 1-5 on five sheets;
- (2) Declaration;
- (3) Assignment and Assignment Cover Sheet;
- (4) Power of Attorney;
- (5) Small-Entity Statement; and
- (6) Our check in the amount of the total filing fee (listed below).

Assistant Commissioner for Patents

January 20, 1999

Page 2

FILING FEE CALCULATION

FOR		Small Entity	Large Entity
Total Claims	20 - 20 = 0	x \$9 = \$ 0.00	or x \$18 = \$
Independent Claims	7 - 3 = 24	x \$39 = \$ 156.00	or x \$78 = \$
Multiple Dependent Claim(s)		+ \$130 = \$ 0.00	or + \$260 = \$
Basic Fee:		+ \$380 = \$ 380.00	or + \$760 = \$
Assignment Recording Fee:	(\$40 per assignee)	+ = \$ 40.00	+ = \$
TOTAL FILING FEES		\$ <u>576.00</u>	\$ <u> </u>

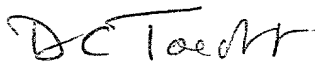
Pursuant to 37 C.F.R. § 1.10 the Applicants request the Patent and Trademark Office to accept this application and accord a serial number and filing date as of the date this application is deposited with the U.S. Postal Service for Express Mail.

If the check is inadvertently omitted, or the amount is insufficient, or should any additional fees under 37 C.F.R. §§ 1.16 to 1.21 be required for any reason in connection with this application, or should an overpayment be included herein, the Assistant Commissioner is authorized to deduct or credit said fees from or to Arnold White & Durkee Deposit Account No. 01-2508/BVEW:154/TOE.

Please date stamp and return the enclosed postcard to evidence receipt of these materials.

Please forward any reply to this communication directly to our Houston office for docketing purposes. The mailing address is P.O. Box 4433, Houston, Texas, 77210-4433; the physical address for courier packages is 750 Bering Drive, Houston, Texas, 77057, and the Houston fax number is 713.787.1440.

Respectfully submitted,



D. C. Toedt III

Reg. No. 31,144

DCT:tyf

Encl:

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Scott H. Hutchinson
Gregory M. Hanka

Group Art Unit: Unknown

Serial No.: Unknown

Examiner: Unknown

Filed: Herewith

Atty. Dkt. No.: BVEW:154/TOE

For: SOFTWARE-IMPLEMENTED METHOD
FOR IDENTIFYING NODES ON A
NETWORK

**VERIFIED STATEMENT (DECLARATION) CLAIMING SMALL ENTITY STATUS
(37 CFR §§ 1.9(F) AND 1.27(C)) - SMALL BUSINESS CONCERN**

I hereby declare that I am

- ☐ the owner of the small business concern identified below:
☒ an official of the small business concern empowered to act on behalf of the concern identified below:

NAME OF SMALL BUSINESS CONCERN: BindView Development Corporation
ADDRESS OF SMALL BUSINESS CONCERN: 5151 San Felipe
Houston, Texas 77056

I hereby declare that the above identified small business concern qualifies as a small business concern as defined in 13 CFR § 121.3-18, and reproduced in 37 CFR § 1.9(d), for purposes of paying reduced fees to the United States Patent and Trademark Office, in that the number of employees of the concern, including those of its affiliates, does not exceed 500 persons. For purposes of this statement, (1) the number of employees of the business concern is the average over the previous fiscal year of the concern of the persons employed on a full-time, part-time or temporary basis during each of the pay periods of the fiscal year, and (2) concerns are affiliates of each other when either, directly or indirectly, one concern controls or has the power to control the other, or a third party or parties controls or has the power to control both.

I hereby declare that rights under contract or law have been conveyed to and remain with the small business concern identified above.

If the rights held by the above identified small business concern are not exclusive, each individual, concern or organization having rights to the invention is listed below* and no rights to the invention are held by any person, other than the inventor, who could not qualify as an

independent inventor under 37 CFR § 1.9(c) if that person made the invention, or by any concern which would not qualify as a small business concern under 37 CFR § 1.9(d), or a nonprofit organization under 37 CFR § 1.9(e).

*NOTE: Separate verified statements are required from each named person, concern or organization having rights to the invention averring to their status as small entities. (37 CFR § 1.27)

FULL NAME: _____

ADDRESS: _____

☐ INDIVIDUAL ☐ SMALL BUSINESS CONCERN ☐ NONPROFIT ORGANIZATION

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR § 1.28(b)).

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

Signature: _____

DATE: 1-19-99

Name: Jeffrey E. Margolis

Title: Vice President, Business Development

66020" 033E260

PATENT

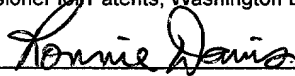
APPLICATION FOR UNITED STATES LETTERS PATENT

for

**Software-Implemented Method
for Identifying Nodes on a Network**

by

Scott H. Hutchinson, Gregory M. Hanka

EXPRESS MAIL MAILING LABEL	
NUMBER	<u>EI 371 159 796 US</u>
DATE OF DEPOSIT	<u>January 20, 1999</u>
I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington D.C. 20231.	
 Signature	

1 **SOFTWARE-IMPLEMENTED METHOD**
2 **FOR IDENTIFYING NODES ON A NETWORK**

3
4 **1. BACKGROUND OF THE INVENTION**

5
6 **1.1. Field of the Invention**

7 The invention was made in attempting to solve a specific problem in connection
8 with “auditing” nodes, e.g., computer workstations and other computers (referred to
9 sometimes here as microcomputers) on a computer network. The problem being ad-
10 dressed by the inventors was that of uniquely identifying nodes on a network for the pur-
11 pose of maintaining a central database reflecting the hardware and software configura-
12 tions of the respective nodes.

13
14 **1.2. Description of Related Art**

15 Recent years have witnessed the development of a category of software applica-
16 tion products which enable a network system administrator (“SYSADMIN”) to track
17 computers and similar equipment (“nodes”), and their components, on computer net-
18 works. Applications in this category are sometimes referred to as “asset management”
19 products.

20 Typically, asset management products assess the hardware and software compo-
21 nents associated with a node on a network and maintain a central database of those nodes
22 and components. The central database is usually remote (on a central computer) from the
23 particular nodes being audited, inventoried or tracked; it is typically used by the
24 SYSADMIN in managing network equipment and software.

25 An illustrative network on which such an asset management product might be
26 used is shown in Figure 1. The network 100 includes two nodes 101, sometimes referred
27 to as “client nodes,” and a node 102, sometimes referred to as a “server node,” connected
28 by communications links 103.

1 The client nodes 101 and server node 102 are typically (but not necessarily) pro-
2 grammable computers. The two depicted client nodes 101 and the server node 102 are
3 merely representative examples; a typical network may include many such nodes.

4 The network 100 may be wide or local in geographical scope, *i.e.*, either a local
5 area network ("LAN") or a ("WAN"). Thus, the client nodes 101 and server node 102
6 may be geographically close, *e.g.*, in the same building, or geographically dispersed, *e.g.*,
7 in separate cities.

8 The network may employ any one of a variety of topologies, protocols, and ma-
9 chine architectures as are known in the art. For example, the network 100 may embody a
10 bus, star, or ring topology, may employ an Ethernet or token-ring protocol, or any other
11 type of network, and may employ a peer-to-peer or client-server architecture.

12 The communications links 103 may be optical fibers, electrical cables, telephone
13 wires, and/or wireless links depending on the particular embodiment implemented.

14 The foregoing examples are mentioned simply for purposes of illustration; those
15 of ordinary skill having the benefit of this disclosure will realize that the network 100
16 may take many other possible conventional forms.

17 The various client nodes 101 typically will either be programmable computers
18 (*e.g.*, a user workstation) or will include one or more programmable processors (*e.g.*, a
19 printer server). As such, each client node 101 will normally include writeable stor-
20 age 104, which may take the form of some or all of, *e.g.*, a floppy disk drive, a hard disk
21 drive, removable storage media (*e.g.*, a ZIP™ drive, JAZ™ drive, a writeable CD-ROM
22 drive, etc.), a tape drive, a flash-memory storage device, or any other suitable storage
23 medium now known or later developed.

24 Like the client node 101, the server node 102 contains a storage 105, *e.g.*, a disk
25 drive.

26 Each client node 101 and the server node 102 contains a network interface card
27 (NIC) 107.

28

1.3. The Desirability of Unique Node Identification

One task of an asset management product is to identify nodes uniquely and to recognize both when nodes 101 have been identified before and when they have not been, so as to recognize the node 101 each time the asset management product 'sees' it in the future, e.g., when the asset management product "audits" the node. This is required in order to match every node 101 up with its records in the central database. This allows the asset management product to know if there have been any changes in the components of a node 101 (e.g., a floppy drive has been removed) since a previous audit.

2. SUMMARY OF THE INVENTION

The invention relates to an asset-management software product, which in one embodiment comprises a server program executing at a server node 102 and a client program executing at a client node 101.

In a first aspect of the invention, at the beginning of each audit, one or more unique attribute values (described in more detail below) of a client node 101 are detected by the client program. These attribute values are transmitted to the server program, which uses them to correlate the client node with a specific record in a central database (creating a new record if necessary). The one or more unique attribute values are also stored to a local database at the client node 101. Upon the next audit, the client program reads information from the local database to find out what the unique attribute values had been during the previous audit and transmits those values as well as the "new" detected values (which may now be different from the previous values). This transmission of out of date information as well as "new" information allows the server program to correctly correlate the client node with its (by now out of date) records in the central database if one or more of these unique attribute values of the node is changed.

In a second aspect of the invention, one specific attribute value tracked by the asset management product is the current address of the network interface card 107 ("NIC address") for each node 101 and any former NIC address it may have had in the past (i.e., its NIC address prior to obtaining the current one) for the purpose of node identification.

1 In a third aspect of the invention, one or more client nodes 101, referred to as
2 “lonely nodes,” either (1) has no active NIC 107 or (2) is configured so that the NIC ad-
3 dress is undetectable by the client program. In the central database, the NIC address for
4 each client node 101 is recorded for use during node identification; for lonely nodes, a
5 fake NIC address is generated and stored. The fake NIC address is created in such a way
6 that it is highly unlikely ever to duplicate any real NIC address in the network in ques-
7 tion.

8 In a fourth aspect of the invention, the local database stored at a client node 101 is
9 duplicated on multiple active partitions of its local hard-disk drive or drives 104, pref-
10 erably on each such partition. Every copy of the local database receives a timestamp re-
11 flecting the time it was last updated, so that subsequent audits of the client node 101 can
12 determine which copy (of possibly many) is the freshest.

13 14 **3. BRIEF DESCRIPTION OF THE DRAWINGS**

15 Other aspects and advantages of the invention will become apparent upon reading
16 the following detailed description and upon reference to the drawings in which:

17 Figure 1 is a block diagram of a hypothetical prior-art network.

18 Figure 2 is a block diagram of a possible variation on such a network.

19 Figure 3 is a before-and-after block diagram of a node identification record in ac-
20 cordance with one implementation of the invention.

21 Figure 4 is a flow chart illustrating some operations performed in accordance with
22 the first and second aspects of the invention described above.

23 Figure 5 shows a hypothetical three-way partitioning of a hard disk drive and re-
24 dundant storage of a node identification record on each partition in accordance with the
25 invention.

26 27 **4. DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS**

28 **4.1 Overview of the Problem**

29 Essential as it is, unique identification in a network 100 is problematic for at least
30 two reasons. First, computers in a typical network 100 are, from the point of view of a

software application, fundamentally alike, with only small differences setting them apart from each other. Second, those small differences are subject to change over the course of time and maintenance. There does not yet exist a standard, ubiquitous ‘fingerprint’ for computers / nodes 101 yet, so asset management products must approximate one using whatever shifting data they can find on each node.

4.1.1 The OEM Motherboard Serial Number as Potential Node Identifier

Referring to Figure 1, at this writing, the closest thing to a fingerprint for microcomputers, and thus for a client node 101, is the new, fledgling standard for motherboard serial numbers. A motherboard 108 is the comparatively large and complex circuit-board on which most of a microcomputer’s electronics are mounted. Of the many components that make up a complete microcomputer, the motherboard 108 is the one best designated the “central component,” in the sense that system administrators expect to be able to change any other component of a microcomputer (such as a hard-drive) and still have their asset management product recognize that it is the same node as before (only now with a new hard-drive). In the current psychology of the art, the motherboard 108 “is” the microcomputer, all else is merely its interface to the world. So, node identification would best track the motherboard 108, via some kind of unique serial-number built into it by its manufacturer.

Unfortunately, as noted above, the standard for motherboard serial-numbers is a fledgling standard. The standard is referred to as “Desktop Management Interface “ or “DMI.” New computers do not necessarily support DMI completely, or support it at all, to say nothing of the world of older computers built before DMI was finalized. So, motherboard serial numbers are not likely for many years to suffice as a universal node identification medium for an asset management product.

4.1.2 The NIC Address as Potential Node Identifier

Again referring to Figure 1, another unique node component for conventional microcomputer-based nodes is the network interface card 107 (“NIC”). As is well known to those of ordinary skill, the term “network interface card” is a generic term for hardware

1 circuitry (usually with appropriate “firmware” programming) that provides an interface
 2 between, e.g., a microcomputer and a network. A network interface card 107 might take
 3 the form of a circuit board that is inserted into an open slot inside a desktop computer or
 4 in a docking station for a notebook computer; or a credit card-sized PCMCIA card (also
 5 known as a PC Card) that can be inserted into a slot in a notebook computer; or circuitry
 6 built directly into the motherboard of either a desktop or notebook computer, i.e., not as a
 7 separate component at all.

8 Each NIC is assigned a unique built-in identifier by its manufacturer, known as
 9 the media access control (“MAC”) address, referred to hereafter as the “NIC address.”
 10 The NIC address is roughly analogous to the vehicle identification number (“VIN”) as-
 11 signed to cars and trucks by their manufacturers. Generally speaking, in this context an
 12 “address” is a sequence of letters, numbers, etc., that serves as an identifier.

13 Referring to Figure 1, suppose that the server node 102 sends out data specifying
 14 that the data is intended for the client node 101 having a NIC 107 whose NIC address is
 15 e.g., “ABC123.” Normally, the data will be broadcast on the network 100. All client
 16 nodes 101 will “hear” the data, including the client node 101 having a NIC 107 whose
 17 NIC address is “QRS789,” but only the specific client node 101 whose address is
 18 “ABC123” will actually respond to the data.

19 The current framework within which NIC addresses are created is administered by
 20 a committee of the Institute of Electrical and Electronics Engineers (“IEEE”). The
 21 committee assigns blocks of addresses to each NIC manufacturer. Each manufacturer
 22 then permanently or semi-permanently installs specific addresses from its respective ad-
 23 dress block(s) into one or more semiconductor chips on the NICs 107 by a process some-
 24 times known as “burning” the address.

25 Ideally, the use of NIC addresses in this manner ensures that every NIC 107 in the
 26 world has a unique address. (In this specification, the term “unique” is used in a local
 27 sense, i.e., unique to a given network. NIC manufacturers sometimes reuse NIC ad-
 28 dresses, but the odds are slim that two identically-addressed NICs 107 would ever show
 29 up in the same network.) Thus, since every computer on a conventional network has a
 30 NIC 107, an asset management product could use the NIC address as its universal node

1 identification, and therefore each computer having a NIC 107 would have a unique ad-
2 dress on the network.

3 The NIC 107, however, is often not a permanent part of a microcomputer's moth-
4 erboard 108; very often it is a removable component plugged into the motherboard.
5 Changing a node's NIC 107 is sometimes required, if for example if the NIC becomes
6 defective, or if the network topology changes so as to require a different type of NIC, or if
7 the node itself is moved (*e.g.*, if a notebook computer is moved from docking station to
8 docking station, as shown in Figure 2, where each docking station contains its own sepa-
9 rate NIC). With a new NIC 107 presumably comes a new NIC address, and thus any as-
10 set management product relying solely on the NIC address for node identification will
11 falter when a node's NIC 107 changes in this way. (By analogy, the FBI would have a
12 similar problem if a person's fingerprints were to change every time the person got a
13 manicure.)

14 15 **4.1.3 The Hard Drive Contents as Potential Node Identifier**

16 Still another potentially unique component in a microcomputer is its fixed disk
17 drive ("hard disk" or "hard drive"), or more precisely, the contents thereof, shown as hard
18 disk storage 104 in Figure 1. During the first inspection or audit of a node, an asset man-
19 agement program can write its tracking data to a hidden file on a node's hard drive 104.
20 During subsequent inspections, the asset management program can retrieve the hidden
21 file and thus recognize the node as the one inspected earlier.

22 The potential downfalls of the hard-drive-contents approach are many, however.
23 For example:

24 1. Hard drives 104 are sometimes "reformatted," in which their entire contents
25 are erased to begin anew; in the process, any hidden files previously placed there by an
26 asset management program are normally lost.

27 2. Hard drives 104 are sometimes moved from one microcomputer (*i.e.*, one cli-
28 ent node 101) to another, which can thoroughly confuse any asset management product
29 that presumes every hard drive 104 to be "married" to its motherboard 108.

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22

6
7
8
9
10

11

12
13
14
15
16

17
18

19
20
21
22

	← FAILURE CASES →									
NODE ID METHOD ↓	Reformat C:	FDISK drive 0	New NIC	NIC swap	Multi- boot	New HDD	HDD swap	Diskless workstation	Cookie- cutter machine	GHOST™ machine
NIC Address	/	/	X	X	/	/	/	/	/	/
OEM serial No.	/	/	/	/	/	/	/	/	/	/
C: serial No.	X	X	/	/	X	X	X	X	X	X
Drive 0 boot record ID	/	X	/	/	/	X	X	X	/	X
Drive 0 firm- ware serial	/	/	/	/	/	X	X	X	/	/
Hidden file on boot drive	X	X	/	/	X	X	X	X	/	/
Hidden file on all drives	X	X	/	/	/	X	X	X	/	/

Table 1: Summary of Some Difficulties with Node-Identifier Approaches

NODE-IDENTIFICATION METHODS: The node-identification methods listed in the far-left column of Table 1 are as follows:

NIC address – as noted above, this is the “MAC address” burned into the firmware of the network interface card 107. It consists of six bytes, three for a vendor code assigned by IEEE and three for a serial number for use by that vendor. Vendors endeavor to avoid duplicating MAC addresses in their production NICs, sometimes even requesting additional vendor IDs from IEEE. In any given installation (network), it is safe to assume that all NIC addresses are unique.

OEM serial number – this is the serial number burned into the motherboard 108 of the workstation / node 101 by its manufacturer. With some difficulty, it can sometimes be changed by a system administrator.

C: serial number – this is the four-byte serial number assigned to a formatted partition on a hard drive 104. It is recreated when the partition is reformatted, but otherwise does not change.

1 Drive 0 boot-record ID – this is a (for all intents and purposes) random number
 2 created in the boot record of the “primary” hard drive 104 on the node’s hard disk con-
 3 troller, as indicated in the BIOS (basic input-output software) of the node 101. In many
 4 computers, the drive 0 boot-record ID is created by an FDISK utility program at the time
 5 that the “partitions” for the hard drive 104 are set up.

6 Drive 0 firmware serial – this is a serial number permanently burned into the ac-
 7 tual hard drive unit 104, and in most cases, it is a very long, very unique string of charac-
 8 ters assigned by the hard drive’s manufacturer. Unfortunately, a few manufacturers do
 9 not bother to use unique serial numbers.

10 Hidden file on boot drive – this is the practice of leaving a hidden file, e.g., an
 11 .INI-type file, on the boot drive 104 of the workstation / node 101, containing node iden-
 12 tification information.

13 Hidden file on all drives – this is the practice of leaving a hidden file, e.g., an
 14 .INI-type file, on every available hard drive 104 on the workstation / node 101 in accor-
 15 dance with the invention, as discussed below.

16
 17 FAILURE CASES: The failure cases listed in the top row of Table 1 are the fol-
 18 lowing:

19 Reformat C: – the primary hard drive partition of the workstation / node 101 was
 20 reformatted, destroying the ‘C: serial number’ and also any hidden files (e.g., .INI-type
 21 files) contained thereon.

22 FDISK drive 0 – a stronger version of ‘Reformat C:’ in which the workstation’s
 23 primary hard drive 104 was repartitioned and reformatted. Not only does this destroy the
 24 ‘C: serial number’ and any hidden files, e.g., .INI-type files, it can also reset the ‘Drive 0
 25 boot-record ID’.

26 New NIC – the workstation received a new network interface card 107, which
 27 gives it a new NIC address.

28 NIC swap – the workstation / node 101 traded network interface cards 107 with
 29 another workstation. Afterward, each workstation / node 101 has the other’s former NIC
 30 address.

Multi-boot – the workstation / node 101 uses boot-manager software (like System Commander™) to boot different operating systems. The various operating systems may designate different partitions as being the “C: drive,” and even more commonly, may regard different partitions as their boot drive. A further complication is that certain partitions may be invisible or inaccessible on a particular operating system; for example, a Windows NT™ NTFS™ partition cannot be accessed by DOS, Windows 95™, Windows 98, Windows 2000, OS/2™.

New HDD – the workstation / node 101 received an entirely new hard drive unit 104. This condition is equivalent to ‘FDISK drive 0’, with the added complication that the ‘Drive 0 firmware serial’ also changes.

HDD swap – the workstation traded hard drives with another workstation. Afterward, each workstation has the other’s former ‘C: serial number’, ‘Drive 0 partition ID’, ‘Drive 0 firmware serial’, and all hidden files, e.g., .INI-type files.

Diskless workstation – the workstation has no local hard drives, and hence, no ‘C: serial number’, no ‘Drive 0 partition ID’, no ‘Drive 0 firmware serial’, and no possibility of any hidden files, e.g., .INI-type files.

Cookie-cutter machines – the workstation was created from a prerecorded image of a hard-drive, including an entire operating system and support software. As a result, it has the same ‘C: serial number’ as all of its siblings.

GHOST™ machines – the workstation / node 101 was created using a PC imaging program such as GHOST™, DiskImage, Disklone, or other automatic software installation programs, which very thoroughly transplant the contents of one workstation’s hard drive 104 to the hard drive 104 of another workstation / node 101. As a result, the new workstation / node 101 has the same “C: serial number” and the same “Drive 0 boot-record ID” as all of its “siblings” created in this way.

4.1.5 Inferences from Analysis

From the information in Table 1, the following may be inferred:

1 “OEM serial number,” i.e., a unique identification number of the motherboard
2 108, is the only 100% reliable node-ID method. Unfortunately, OEM serial-number de-
3 tection is not yet widely available, and is far from an industry standard.

4 “NIC address” is an excellent alternative node-ID method, if only movements of
5 NICs 107 could be handled somehow.

6 If “NIC address” could somehow be combined with “hidden file [e.g., .INI-type
7 file] on all drives”, the result would be 100% reliable for any single failure condition.
8 (The case of multiple coincident failures is likely to be too complex to handle with any-
9 thing other than the “OEM serial” approach.)

11 **4.2. Illustrative Software-Based Solution**

12 The multi-faceted approach of the invention is explained with reference to the
13 network 100 shown in Figure 1.

15 **4.2.1 Client Program; Server Software**

16 The software running in each client node 101 includes an “agent” program re-
17 ferred to sometimes as an “audit” program and referred to here as a client program. The
18 client program may be designed to run as a conventional foreground program, or as a
19 background application, in whatever form is appropriate for the operating system in
20 question (e.g., a terminate-and-stay-resident [TSR] program under MS-DOS or PC DOS,
21 or a background service under other operating systems). Some well-known operating
22 systems at this writing include, e.g., Windows 3.1; Windows 95; Windows 98; Win-
23 dows NT; Windows 2000; Mac OS; various flavors of UNIX; and the like.

24 The client program exchanges information, via the communications links 103,
25 with a server program that is likewise running on the server node 102. The server pro-
26 gram performs many of the functions described below.

27 (In this specification, phrases such as “the client program doing X,” where X is
28 some function or functions, will be understood by those of ordinary skill as referring to
29 one or more programmable processors performing the specified function(s) under control
30 of the software in question.)

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30

5

6
7
8
9
10
11
12
13
14

15
16
17
18
19
20

21
22
23
24
25
26
27
28
29
30

poses in case no NIC is detected by the client program. It will be appreciated that any number of former NIC addresses may be stored in the node-identification record 305 if desired, thus creating a history for that particular client node 101.

The node-identification record 305 may be initialized by the client program when that software runs on the client node 101 for the first time. It may be updated either on a scheduled basis or in response to specific events (e.g., every time the client program is “booted up,” i.e., started, or every time the client program performs an audit of the client node 101).

4.2.3 Central Database at Server Node

A central database (not specifically shown in the drawing figures) is stored in the storage 105 at the server node 102. Generally speaking, the database is a compilation or some or all of the information stored in the node-identification records 305 at the respective client nodes 101, typically with one data record in the database per client node 101. The database may also store other audit-related information provided by a client node 101, again typically in one record per node. The database is conventionally initialized and periodically updated, either on a regular scheduled basis or in response to specific events.

4.3 Basic Node Identification Method

A node’s NIC address represents a reliable client node 101 identification method. The caveat, however, is that network interface card movements must be tracked somehow. The node-identification records 305 and the central database provide tools that can be used in such auditing.

4.3.1 Initial Audit

During an initial audit of a client node 101, the client program running on that client node conventionally detects the node’s NIC address. The just-detected NIC address is then stored in a new node-identification record 305 (e.g., a new, timestamped .INI-type file) at the client node’s data storage 104 as described in Section 4.2.2 above. If a NIC

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30

6

7
8
9
10
11
12
13
14
15

16

17

18

20

21

22

23

24

25
26
27
28
29
30

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

9
10
11
12
13
14

15
16
17
18
19
20
21
22

23
24
25
26
27

28
29
30

1 A pseudocode appendix setting forth the server-program algorithm in detail is re-
2 produced as Appendix 1 below.

4 **4.4 Replicated Node-Identification Records**

5 It was noted above that a local database is maintained at the data store 104 of each
6 client node 101. The local database contains a description of some of that client node's
7 unique attribute values that may be used for identification.

8 Referring to Figure 5 as a hypothetical example: In one aspect of the invention,
9 when the data store 104 includes one or more hard disks or similar partitionable storage
10 media, a mirrored copy of the local database is maintained on each active partition 505,
11 510, 515, etc., of each such hard disk. At the beginning of an audit, the client program
12 checks the respective internal timestamps of all accessible copies of the local database
13 505, 510, 515, etc., to determine which copy is most recent. The most recent copy is
14 utilized during the audit as described above. Afterward, the updated local database is re-
15 written onto the accessible partitions 505, 510, 515, etc., of all hard drives 104, overwrit-
16 ing any old copies. Included in the rewrite is an update of the timestamps of the copies.

17 The aforementioned functionality permits the asset management product to deal
18 with nodes that "boot" with multiple operating systems. Sometimes, a computer will
19 have several operating systems installed, and will boot between them at-will. Local hard-
20 disk partitions 505, 510, 515, etc., that are visible under one operating system are some-
21 times invisible under other operating systems. So, an audit under one operating system
22 may generate six copies of the local database, but then a subsequent audit under another
23 operating system may update only three of them. The next audit under the original op-
24 erating system will show three old copies and three new copies. Thanks to the times-
25 tamps, the client program can tell which copy or copies contain the latest information
26 about the node's unique attribute values. This information is used by the central database
27 for node identification.

28 In the hypothetical example shown in Figure 5, the storage 104 at the client
29 node 101 is configured with a hard disk drive that is divided into separate partitions 505,
30 510, 515, etc., that may be logically configured as drives C, D, and E respectively, with

different, selectively-bootable operating systems on each partition. In the illustration of Figure 5, the three logical drives are shown as being bootable into MS-DOS, UNIX, and Windows NT respectively.

Such a partitioning can cause complications for the audit process. Suppose that the client node 101 is “booted up” (i.e., started) into, say, MS-DOS on drive C, but the node-identification record 305 is stored only on drive D containing, say, the UNIX operating system. It will be quickly recognized by those of ordinary skill having the benefit of this disclosure that, because of certain limitations in MS-DOS, any files stored on drives D and E – which could include the node-identification record 305 – may be inaccessible to client program running under MS-DOS unless the CONFIG.SYS file for MS-DOS is properly configured. (It will be apparent that client program appropriate for the operating system actually booted must be provided on an accessible logical drive.)

In such a configuration, the node-identification record 305 may be replicated, i.e., redundantly stored within each of the logical drives C, D, and E, thus making its contents available to client program no matter which operating system is booted. Consequently, if the client node 101 is booted into MS-DOS from drive C, and drives D and E are inaccessible, the copy of the node-identification record 305 can still be updated by client program running under MS-DOS; the updated copy is identified by reference numeral 310.

As noted above, however, that in turn presents another issue: If only the copy 310 of the node-identification record on drive C is updated (because by hypothesis drives D and E are inaccessible), then the replicated copies of the node-identification record 305 on drives D and E may be out of date the next time that UNIX or Windows NT is booted. This issue may be addressed by having the client program, upon its own booting, check the timestamps in any accessible copies of the node-identification record 305 and 310 and use only the latest one as described above.

(As used in this specification, including in the claims, the term “redundantly stored” is not intended to be limited to the situation in which all instances, i.e., all copies of the node-identification record, have been synchronized. That is, the term “redundant storage” is intended to include, not exclude, the situation in which one copy of the record has been updated but the other copies have not yet been updated, as shown in Figure 3.)

4.5 Tracking of “Lonely” Nodes

It is not always possible to detect a network interface card 107 on certain client nodes 101, thus precluding the use of a NIC address as the exclusive node identifier for that client node 101. For example, when a client node 101 that is part of a Windows NT™ network 100 is booted under DOS, it is likely that the network interface card 107 will not be active if no DOS drivers for the network interface card 107 are installed. Or, referring again to Figure 2, an undocked notebook computer 200 may have its network interface card 107 located in a docking station 201; the notebook computer 200 is considered part of the network 100, but it does not have an active network interface card 107 because it is undocked. Other client nodes 101 in the network might not have a network interface card 107 at all, but they are still “in the network” 100, so to speak, from the perspective of the system administrator, who desires to be able to keep track of them. Such client nodes 101 are referred to as “lonely nodes.” Without a NIC address, these nodes do not have the universal ‘fingerprint’ required by the asset management product for the recognition purposes.

So, when initially building a node-identification record 305 for a “lonely” client node 101 (i.e., one without an active network interface card 107), the client program generates, and stores in the node-identification record 305, a “fake” NIC address to correspond only to that particular lonely node. The generated NIC address is fake in the sense that it is unique to the network 100.

The fake NIC address is used by the client program and the server program in the same general way as a real NIC address, unless and until a real NIC address is subsequently detected on the node in question. If a real NIC address is subsequently detected on the node, then the fake NIC address is retired as the node’s “former NIC address.”

Fake NIC addresses must not duplicate any possible real NIC address, lest a fake NIC address accidentally duplicate the address of a real NIC on the network and thus confuse the two nodes in the central database. Non-duplication is accomplished by using a block of NIC addresses allocated to the applicant by IEEE. The asset management prod-

uct can create fake NIC addresses anywhere within this block under IEEE's guarantee that no NIC manufacturer can be assigning NIC addresses in the same range.

In one implementation, the fake NIC address is generated by combining a known signature portion (*e.g.*, a three byte signature code or NIC vendor ID assigned to the software vendor by the IEEE) with a pseudorandomly generated portion, for a total of six bytes of data. A typical fake NIC address looks like this: 00-90-D4-1F-E3-22. The first three bytes of the fake NIC address consist of a NIC vendor ID assigned by the IEEE. The last three bytes are generated pseudorandomly by the asset management product. An example of an algorithm for generating pseudorandom portions is shown in Appendix 3.

The signature portion of the fake NIC address is included so that the server software running on the server 102 will recognize that the NIC address was artificially generated. This provides the system administrator with greater node inventory reliability because the asset management product knows not to report the NIC address as a genuine one. The pseudorandom portion is added to the fake NIC address 24 in case the network 100 has more than one lonely client node 101.

It will be appreciated by those skilled in the art that a conventional pseudorandom-number generator can be used to generate the pseudorandom portion of the fake NIC address. The use of a pseudorandom-number generator provides a reasonable assurance that the fake NIC address will be unique within the network 100. One algorithm for generating fake NIC addresses is set out in Appendix 3 below. Because each byte has 256 possible values, this algorithm yields $256 \times 256 \times 256 = 16,777,216$ different possible fake NIC addresses, which means the chance of getting duplicate fake NIC addresses is acceptably small.

An additional advantage provided by the use of fake NIC addresses is that it permits other existing asset management applications software to continue operating normally without the system administrator having to worry about whether a client node 101 actually has a NIC address.

Upon subsequent audits of the network 100, assuming that the lonely node is eventually connected to the network via a network interface card 107, the lonely node's actual NIC address will eventually be detected and the respective records will be updated

1 in the node-identification record 305 at the client node itself and in a record at the data-
2 base at the server node 102. As noted above in this Section, the fake NIC address is then
3 retired as the node's now-former NIC address.

4.6 Some Alternative Implementations

6 In hindsight, it will be appreciated by those of ordinary skill having the benefit of
7 this disclosure that the node identification technique disclosed here can be used in a vari-
8 ety of situations.

9 For example, the node identification technique can be used any time information
10 is to be transmitted from a client node 101 to one or more other nodes, whether or not in
11 response to a query by the other node(s). For example, in hindsight it will be apparent
12 that the client node 101 may be programmed automatically to send reports of various
13 kinds to another node running appropriate server software, without waiting for an audit
14 command or other query from the other node.

15 As another example, the basic approach in which the client program sends out
16 both current and former identification information to a server program may be used in
17 contexts not involving a NIC address, e.g., over the Internet.

18 As still another example, the client program may be designed to perform some of
19 the functions of the server program, thus possibly freeing up the server program and its
20 host computer from operations that can be performed at the client program. This permits
21 the asset management product to operate on a standalone basis to that extent, with the
22 relevant portions of the database 204 being maintained (or replicated) at the local stor-
23 age 104.

4.7 Program Storage Device

26 It will be apparent to those of ordinary skill having the benefit of this disclosure
27 that the client program and the server software may be implemented by programming one
28 or more suitable general-purpose computers having appropriate hardware. The pro-
29 gramming may be accomplished through the use of one or more program storage devices
30 readable by the computer and encoding one or more programs of instructions executable

1 by the computer for performing the operations described above. The program storage
2 device may take the form of, e.g., one or more floppy disks; a CD ROM or other optical
3 disk; a magnetic tape; a read-only memory chip (ROM); and other forms of the kind well-
4 known in the art or subsequently developed. The program of instructions may be "object
5 code," i.e., in binary form that is executable more-or-less directly by the computer; in
6 "source code" that requires compilation or interpretation before execution; or in some in-
7 termediate form such as partially compiled code. The precise forms of the program stor-
8 age device and of the encoding of instructions is immaterial here.

9 10 **5. ORDER OF OPERATIONS IN METHOD CLAIMS**

11 Some of the claims below recite the performance of certain operations or func-
12 tions. It will be understood, by those of ordinary skill having the benefit of this disclo-
13 sure, that the operations or functions in question are not necessarily required to be per-
14 formed in the specific order in which they are listed in the claims.

15 16 **6. SOFTWARE PSEUDOCODE APPENDIXES**

17 The appendixes below are pseudocode listings for a specific implementation of
18 the invention by the assignee in client program and server software.

Appendix 1: General Server Program Node-Identification Algorithm

```

1      if (local database contained a previous NIC address)
2
3      {
4          if (central DB has node with same current and previous NIC addresses)
5          {
6              //
7              // Central database already aware of the NIC change
8              //
9              Audit as existing node.
10         }
11         else if (central DB has node with same previous NIC address and same bios date)
12         {
13             //
14             // NIC change or NIC swap since last audit; follow local database
15             //
16             Audit as existing node.
17             Update central database with new NIC address.
18         }
19         else if (central DB has a node with same current address and same bios date)
20         {
21             //
22             // HDD swap, local database is from another node; follow current NIC instead
23             //
24             Audit as existing node.
25         }
26         else
27         {
28             Insert as new node.
29         }
30     }
31 else
32 {
33     //
34     // No local database found on the node
35     //
36     if (central DB has a node with same current NIC address and same bios date)
37     {
38         //
39         // Local database lost; follow current NIC
40         //
41         Audit as existing node.
42     }
43     else
44     {
45         //
46         // Node has not been audited before
47         //
48         Insert as new node.
49     }
50 }
51
52

```

Appendix 2: Start-to-Finish Audit Algorithm

```

1
2
3
4  ****AGENT****
5
6  Try to detect the node's 'OEM serial number'...
7      * Compaq BIOS call
8      * DMI call
9
10 Search for INI file(s) written during previous audit.
11 if (any INI files found)
12     {
13         Retrieve 'former NIC address' as it was during the previous audit.
14         Note: INI files are timestamped so that we know which one is newer.
15     }
16
17 Try to detect 'current NIC address'...
18     * IPX via Winsock
19     * direct IPX call
20     * NetBIOS call
21     * VINES call
22     * request GUID from Windows
23     * search Windows registry
24     * ask local Novell server
25
26 if (no 'current NIC address' detected or found from previous audit)
27     {
28         if (one or more local fixed disks are available to hold INI files)
29             {
30                 Generate a random 'current NIC address' for use until the real NIC address is detected.
31             }
32     }
33
34 Create a "audit start request" message, containing (among other things):
35     * current NIC address (or the temporary address if none)
36     * former NIC address (from INI file, if any)
37     * OEM serial number (if any)
38
39 Send the "audit start request" message to the server.
40
41 ****SERVER****
42
43 Try to detect the node's NIC address from inside the server, by examining the node's NetWare
44 connection.
45 if (success)
46     {
47         Discard the agent-detected 'current NIC address' in favor of that detected in the server.
48     }
49
50 Identify the node...
51     {
52         if (auditing a NetWare file server)
53             {

```

```

54         if (database has a file-server node with the same name)
55         {
56             // A server-node is identified strictly by its node-name, as opposed to the
57             // by its OEM serial no. or /NIC-address. This is because where file-servers
58             // are concerned, the name *is* a unique identifier.
59         }
60     else
61     {
62         //
63         // Auditing a regular workstation
64         //
65         if (OEM serial number at least five characters long was detected)
66         {
67             if (OEM serial found in database)
68             {
69                 Audit as existing node.
70             }
71         }
72         else if (NIC address available)
73         {
74             if (hidden files contained a previous node address)
75             {
76                 if (database has a node with same current and previous address)
77                 {
78                     //
79                     // Servers are already aware of the NIC change
80                     //
81                     Audit as existing node.
82                 }
83                 else if (database has a node with same previous address and same bios date)
84                 {
85                     //
86                     // NIC change or NIC swap since last audit; follow hidden files
87                     //
88                     Audit as existing node.
89                     Update node address.
90                 }
91                 else if (database has a node with same current address and same bios date)
92                 {
93                     //
94                     // HDD swap; follow current NIC
95                     //
96                     Audit as existing node.
97                 }
98             }
99             else
100             {
101                 Insert as new node.
102             }
103         }
104         else
105         {
106             //
107             // No hidden files found
108             //

```

```

108         if (database has a node with same current address and same bios date)
109             {
110                 //
111                 // HDD reformat and ini files lost; follow NIC
112                 //
113                 Audit as existing node.
114             }
115         else
116             {
117                 //
118                 // Node has not been audited before
119                 //
120                 Insert as new node.
121             }
122     }
123 }
124 else
125 {
126     //
127     // No NIC, no local fixed drives; must be a lonely audit
128     // Here, the console must inject a node address into the rawfile
129     // before uploading it, unless the rawfile
130     // is to be identified by node-name only (a risky venture)
131     //
132 }
133 }
134 }
135
136 Send an "audit start reply" message back to the agent.
137 * The message includes the node's server-detected NIC address, if any.
138
139 ****AGENT****
140
141 Receive the "audit start reply" message from server.
142
143 if ("audit start reply" message contains a 'current NIC address' as detected by the server)
144 {
145     Discard any agent-determined 'current NIC address'
146     in favor of the server-determined 'current NIC address'.
147 }
148
149 if (one or more local fixed disks are available to hold INI files)
150 {
151     if (any INI files found)
152     {
153         if (INI file 'current NIC address' is different from the new 'current NIC address')
154         {
155             Retire INI file 'current NIC address' slot to the 'former NIC address' slot.
156             Record 'current NIC address' to the INI file 'current NIC address' slot.
157             Refresh INI file(s) with the current date and time.
158         }
159     }
160     else
161     {
162         The NIC address(es) recorded in the INI file(s) are still accurate.

```

```
162         Refresh INI file(s) with the current date and time.
163     }
164 }
165 else
166 {
167     Record 'current NIC address' to new INI file(s) for use during future audits.
168 }
169 }
```


Appendix 3: Fake NIC Address Generation Algorithm

```
1
2
3 void GenerateFakeNICAddress(U8 address[6])
4 {
5     //
6     // Create a random NIC address for temporary use by a node that
7     // cannot currently detect its own NIC address
8     //
9     // First three digits are our NIC address block, also known as
10    // the ethernet vendor code.
11    // 00-90-D4 is NETinventory's official address block as
12    // assigned by IEEE on 06/24/1998.
13    //
14    address[0] = 0x00;
15    address[1] = 0x90;
16    address[2] = 0xD4;
17    //
18    // seed random number generator
19    //
20    srand((unsigned int)(time(NULL)));
21    //
22    // last three bytes of NIC address are random digits
23    //
24    address[3] = (U8)(rand() % 256);
25    address[4] = (U8)(rand() % 256);
26    address[5] = (U8)(rand() % 256);
27 }
28
```

WHAT IS CLAIMED IS:

1 1. A method, executed by a node on a network, of transmitting identifying infor-
2 mation about the node, the method comprising:

3 (a) determining a current node identifier value;

4 (b) retrieving, from a data storage at the node, a former node identifier
5 value for the node; and

6 (c) transmitting the current node identifier value and the former node
7 identifier value.

1 2. The method of claim 1, wherein (1) the value of the node identifier for any par-
2 ticular node in the network is dependent on one or more node-identification attributes of
3 that node, and (2) determining the current node identifier value includes an attempt to
4 detect the then-current values of said one or more node-identification attributes.

1 3. The method of claim 2, wherein the attempt to detect said one or more node-
2 identification attributes fails to detect at least one of said node-identification attributes,
3 and further comprising (i) retrieving, from a data storage at the node, a stored value con-
4 taining the result of a past live detection of the said one or more node-identification at-
5 tributes, referred to as a previously-detected node identifier value; and (ii) transmitting
6 the previously-detected node identifier value.

1 4. The method of claim 1, wherein (i) the node includes a network interface card,
2 and (ii) the node identification information includes a network interface card value, re-
3 ferred to as a NIC address value.

1 5. The method of claim 4, wherein the NIC address value comprises a signature
2 portion and a pseudorandomly generated portion.

1 6. The method of claim 1, wherein the former node identifier value is redundantly
2 stored in multiple partitions within the data storage at the node.

1 7. The method of claim 6, wherein (x) each copy of the former node identifier
2 value is associated with a timestamp, and (y) retrieving the former node identifier value
3 comprises retrieving the respective copy associated with the most recent timestamp.

1 8. A method, executed by a server node on a network, for recording, in a data-
2 base, information about a client node, comprising:

3 (a) receiving information from the client node, said information including
4 node-identification information for the client node that includes (i) a current node-
5 identifier value, and (ii) a former node-identifier value; and

6 (b) storing, in a record in the database associated with the node-
7 identification information, the current node-identifier value and the former node-identifier
8 value.

1 9. The method of claim 8, wherein each of the current node-identifier value
2 and the former node-identifier value is a NIC address value.

1 10. The method of claim 9, wherein the NIC address value comprises a signa-
2 ture portion and a pseudorandomly generated portion.

1 11. A program storage device readable by a processor in the node of a speci-
2 fied one of claims 1 through 7 and encoding a program of instructions including instruc-
3 tions for performing the operations recited in the specified claim.

1 12. A program storage device readable by a processor in the server node of a
2 specified one of claims 8 through 10 and encoding a program of instructions including
3 instructions for performing the operations recited in said specified claim.

1 13. In a node on a network, a data store comprising a machine-readable data
2 structure accessible to a processor in the node and containing node-identification infor-
3 mation for the client node that includes (i) a current node-identifier value, and (ii) a for-
4 mer node-identifier value.

1 14. The data store of claim 13, wherein each of the current node-identifier
2 value and the former node-identifier value is a NIC address value.

1 15. The data store of claim 14, wherein the NIC address value that constitutes
2 the current node-identifier value includes a signature portion and a pseudorandomly gen-
3 erated portion.

- 1 16. In a node on a network, a data store comprising:
2 (a) a plurality of machine-readable data structures accessible to a processor
3 in the node;
4 (b) each said data structure containing node-identification information for
5 the client node that includes (i) a current node-identifier value, and (ii) a former node-
6 identifier value.
7 (c) each said data structure being associated with a timestamp.

- 1 17. The data store of claim 16, wherein the current node-identifier value is a
2 NIC address value.

- 1 18. The data store of claim 17, wherein the NIC address value comprises a
2 signature portion and a pseudorandomly generated portion.

- 1 19. In a server node on a network. that includes a client node, a machine-
2 readable data structure accessible to a processor in the server node, comprising a current
3 NIC address value for the client node and a former NIC address value for the client node.

- 1 20. The machine-readable data structure of claim 19, wherein the current NIC
- 2 address value comprises a signature portion and a pseudorandomly generated portion.

continued

ABSTRACT

A method and apparatus for a node on a network to transmit identifying information about itself, typically in the course of an audit of the hardware and/or software that are present on the network. The method involves the node transmitting both a current node identifier value and a former node identifier value, each typically a network interface card (NIC) address. The former node identifier value permits the receiver of the transmission to determine which node is involved even if the current node identifier has been changed since the previous audit (e.g., because of a change of NICs). The current and former node identifier values may be stored in a timestamped hidden file, e.g., an .INI-type file. Such file may be redundantly stored on multiple partitions, with the timestamp used to determine which is the most recently updated file.

FIG. 1
(PRIOR ART)

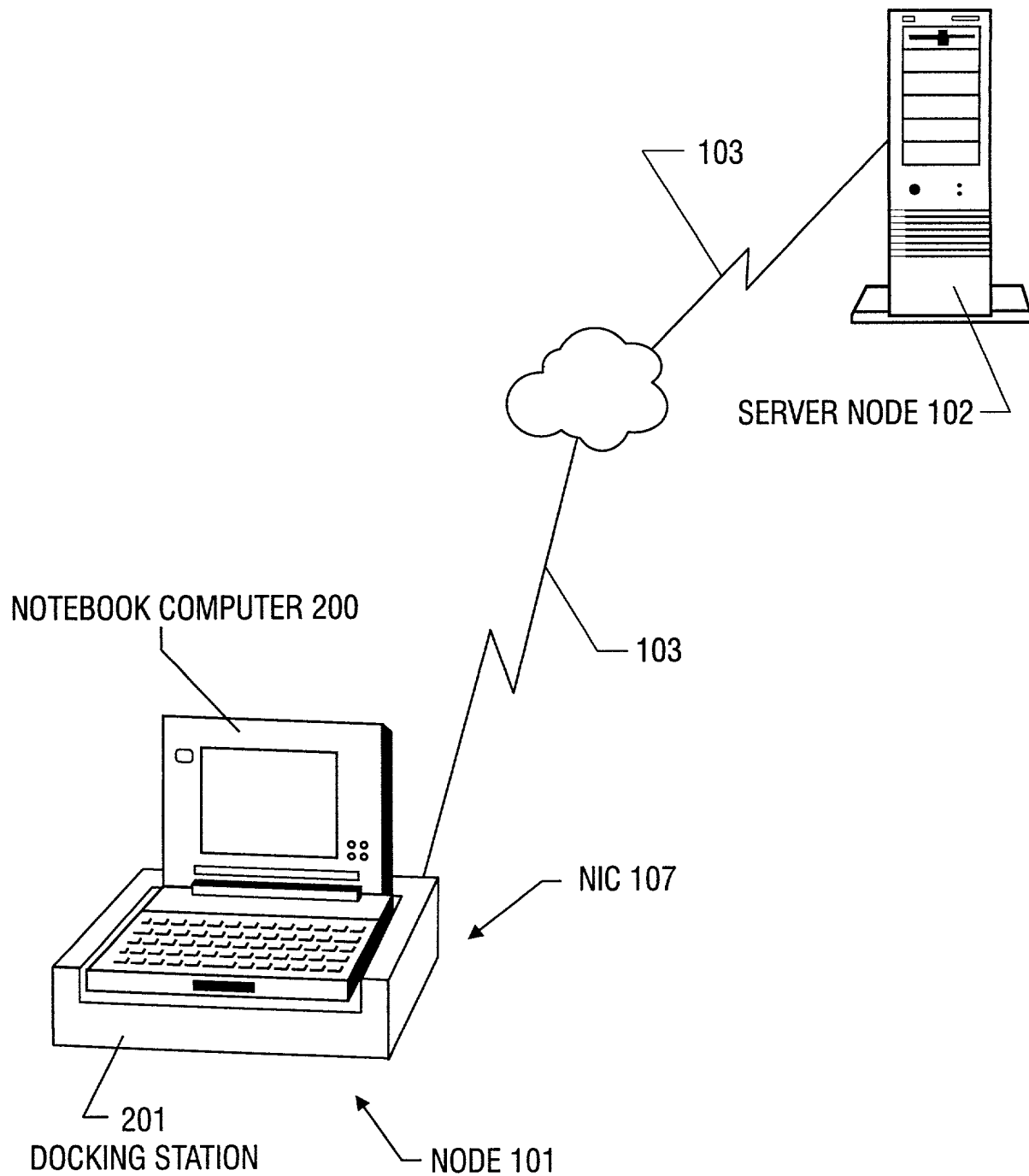
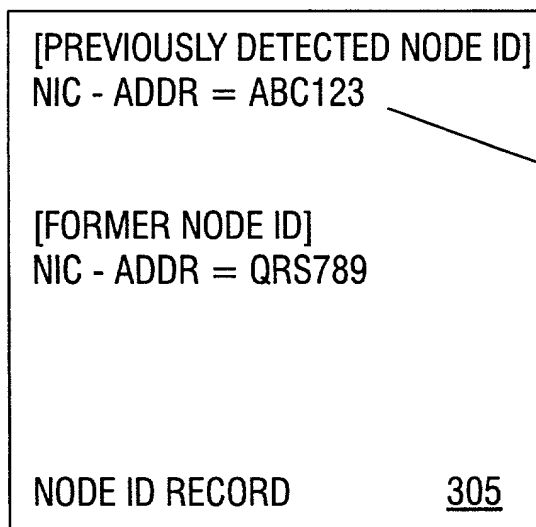


FIG. 2
(PRIOR ART)

BEFORE DETECTING
NIC ADDRESS CHANGE
TO "DEF456"



UPDATED AFTER DETECTING
NIC ADDRESS CHANGE
TO "DEF456"

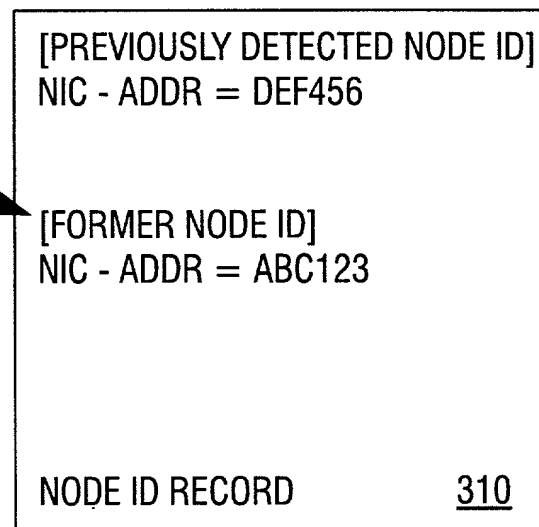


FIG. 3

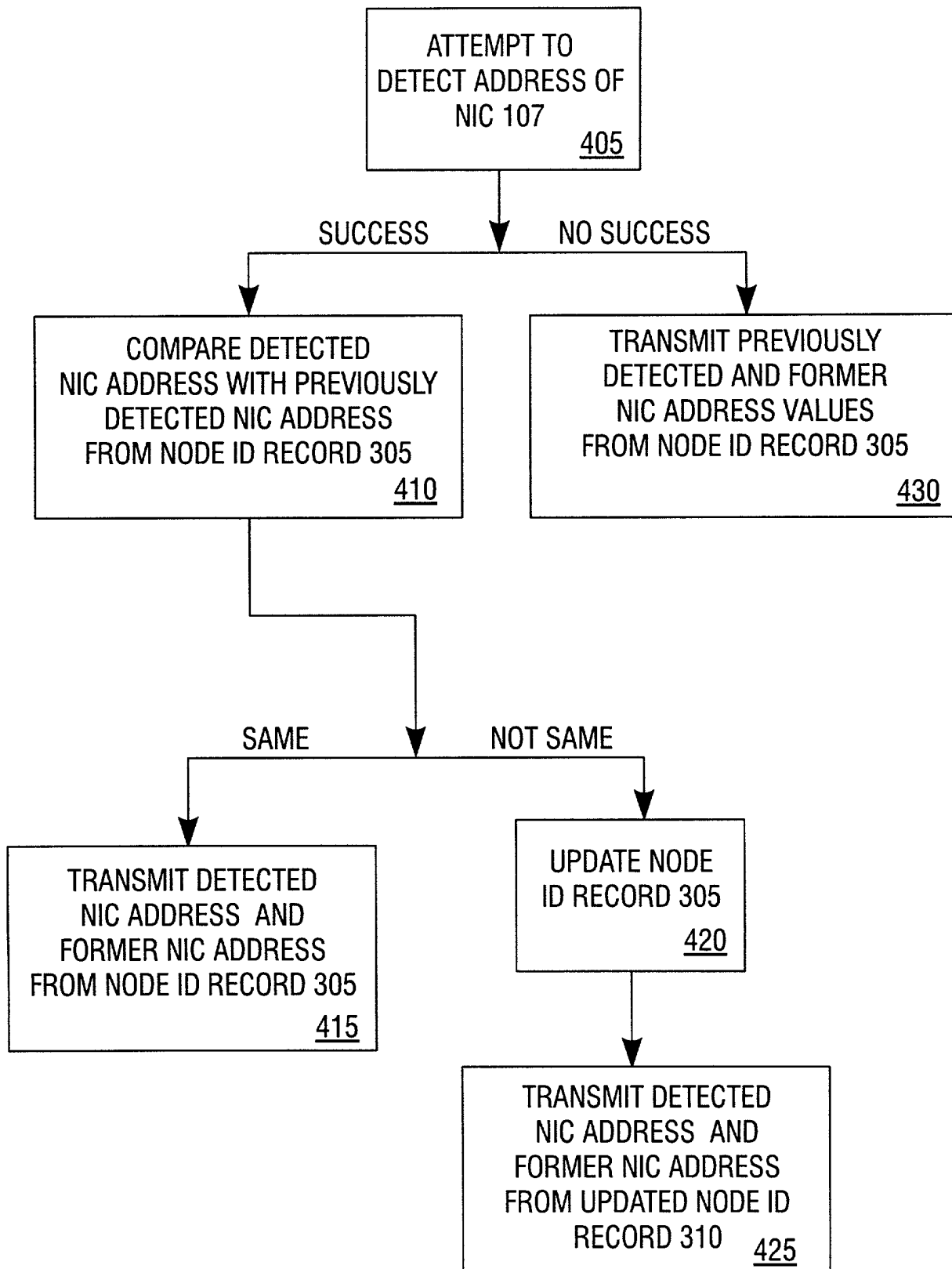


FIG. 4

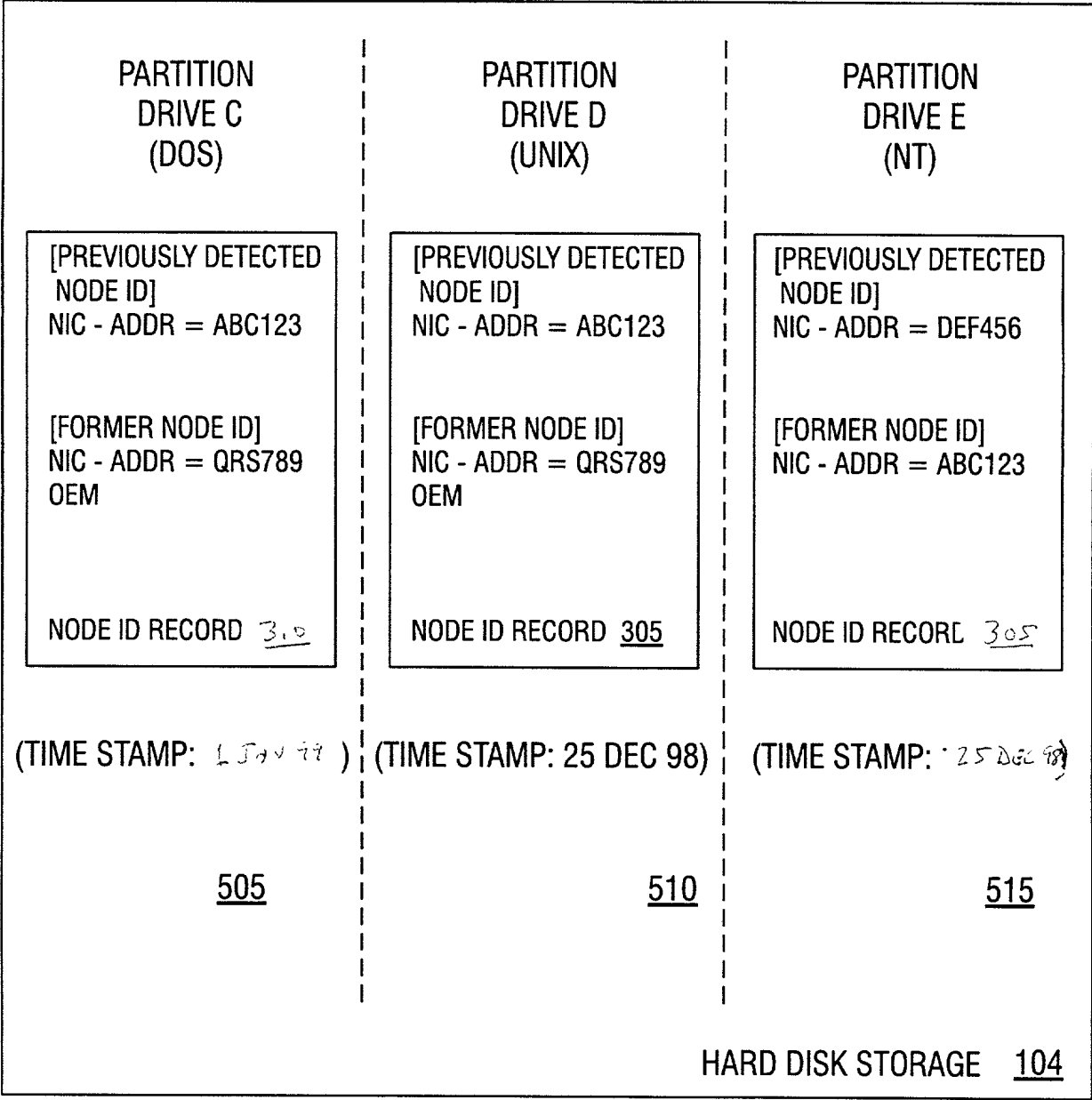


FIG. 5

DECLARATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or the below named inventors are the original, first and joint inventors (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled **Software-Implemented Method for Identifying Nodes on a Network**, the Specification of which:

- ☒ is attached hereto.
☐ was filed on _____ as Application Serial No. _____.

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims.

I acknowledge the duty to disclose to the Patent and Trademark Office all information known to me to be material to patentability of the subject matter claimed in this application, as "materiality" is defined in Title 37, Code of Federal Regulations, § 1.56.

I hereby claim priority benefits under Title 35, United States Code, § 119 of any foreign application(s) for patent, United States provisional application(s), or inventor's certificate listed below and have also identified below any foreign application for patent, United States provisional application, or inventor's certificate having a filing date before that of the application on which priority is claimed:

PRIORITY APPLICATION(S)			Priority Claimed
(Number)	(Country)	(Date Filed)	Yes/No
(Number)	(Country)	(Date Filed)	Yes/No

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose all information known to me to be material to patentability of the subject matter claimed in this application, as "materiality" is defined in Title 37, Code of Federal Regulations, § 1.56, which


become available between the filing date of the prior application and the national or PCT international filing date of this application:

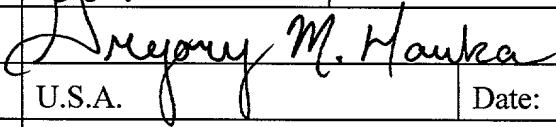
(Application Serial No.)	(Filing Date)	(Status)
--------------------------	---------------	----------

(Application Serial No.)	(Filing Date)	(Status)
--------------------------	---------------	----------

I hereby direct that all correspondence and telephone calls be addressed to D. C. Toedt, Arnold White & Durkee, P.O. Box 4433, Houston, Texas 77210, (713) 787-1400.

I HEREBY DECLARE THAT ALL STATEMENTS MADE OF MY OWN KNOWLEDGE ARE TRUE AND THAT ALL STATEMENTS MADE ON INFORMATION AND BELIEF ARE BELIEVED TO BE TRUE; AND FURTHER THAT THESE STATEMENTS WERE MADE WITH THE KNOWLEDGE THAT WILLFUL FALSE STATEMENTS AND THE LIKE SO MADE ARE PUNISHABLE BY FINE OR IMPRISONMENT, OR BOTH, UNDER SECTION 1001 OF TITLE 18 OF THE UNITED STATES CODE AND THAT SUCH WILLFUL FALSE STATEMENTS MAY JEOPARDIZE THE VALIDITY OF THE APPLICATION OR ANY PATENT ISSUED THEREON.

Inventor's Full Name:	Scott	H.	Hutchinson
Inventor's Signature:			
Country of Citizenship:	U.S.A.	Date:	1/20/99
Residence Address: (street, number, city, state, and/or country)	4702 Colchester Way Missouri City, Texas 77459		
Post Office Address: (if different from above)			

Inventor's Full Name:	Gregory	M.	Hanka
Inventor's Signature:			
Country of Citizenship:	U.S.A.	Date:	01/20/1999
Residence Address: (street, number, city, state, and/or country)	1715 Bowline Road Houston, Texas 77062		
Post Office Address: (if different from above)			

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Scott H. Hutchinson

Gregory M. Hanka

Group Art Unit: Unknown

Serial No.: Unknown

Examiner: Unknown

Filed: Herewith

Atty. Dkt. No.: BVEW:154/TOE

For: SOFTWARE-IMPLEMENTED METHOD
FOR IDENTIFYING NODES ON A
NETWORK

**ELECTION UNDER 37 C.F.R. §§ 3.71 AND 3.73
AND POWER OF ATTORNEY**

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

The undersigned, being Assignee of record of the entire interest in the above-identified application by virtue of an assignment recorded in the United States Patent and Trademark Office as set forth below, hereby elects, under 37 C.F.R. § 3.71, to prosecute the application to the exclusion of the inventors.

The Assignee hereby revokes any previous Powers of Attorney and appoints:

Hugh R. Kress, PTO Reg. No. 36,574; James C. Pistorino, PTO Reg. No. P 44,290; Victor H. Segura, PTO Reg. No. P-44,329; D. C. Toedt, PTO Reg. No. 31,144; and J. Paul Williamson, PTO Reg. No. 29,600.


each an attorney or agent of the firm of ARNOLD WHITE & DURKEE, as its attorney or agent for so long as they remain with such firm, with full power of substitution and revocation, to prosecute the application, to make alterations and amendments therein, to transact all business in the Patent and Trademark Office in connection therewith, and to receive any Letters Patent, and for one year after issuance of such Letters Patent to file any request for a certificate of correction that may be deemed appropriate.

Pursuant to 37 C.F.R. § 3.73, the undersigned has reviewed the evidentiary documents, specifically the Assignment to BindView Development Corporation, referenced below, and certifies that to the best of my knowledge and belief, title remains in the name of the Assignee.

Please direct all communications as follows:

D. C. Toedt
ARNOLD WHITE & DURKEE
750 Bering Drive
Houston, Texas 77057
(713) 787-1400

ASSIGNEE:
BindView Development Corporation

By: 
Name: Jeffrey E. Margolis
Title: Vice President
Business Development

Date: 1-19-99

ASSIGNMENT:

- ☒ Concurrently filed
☐ Previously recorded

Date:

Reel:

Frames: